Proceedings of the 1993 IEEE International Conference on Robotics and Automation
May 2-7, 1993, Atlanta, Georgia

Laserradar and Sonar Based World Modeling and Motion Control for Fast Obstacle Avoidance of the Autonomous Mobile Robot MOBOT-IV

Markus Buchberger, Klaus-Werner Jörg, Ewald von Puttkamer

Computer Science Department • Robotics Research Group
Kaiserslautern University, PO Box 3049, 67653 Kaiserslautern, Germany
e-mail: joerg@informatik.uni-kl.de

Abstract

One fundamental property a really useful autonomous mobile robot has to be equipped with is the capability to effectively avoid collisions in realtime. The focus of this paper is to describe the obstacle avoidance mechanism of MOBOT-IV*, which serves as a test-platform within the scope of our research project on autonomous mobile robots for indoor applications. The mechanism utilizes heterogenous information obtained from a laserradar and a sonar senor system in order to achieve a reliable and complete world model for realtime collision avoidance. This world model is transformed by a motion control component into a virtual position deviation to perform obstacle avoidance by using a track-control algorithm.*

1. Introduction

An Autonomous Mobile Robot (AMR) is a mobile system capable of interpreting, planning and executing a given task without any external support. Therefore, an AMR must be able to explore an a priori unknown environment systematically in order to perform some useful, goal-oriented operation after the exploration phase is finished. During this exploration phase the robot's perception system has to incrementally build up internal models of the environment which most accurately represent topological and geometrical properties of the environment. Consequently, these world models are the basis for goal-oriented behaviour. On the other hand it is absolutely necessary that whatever action the robot performs it avoids to collide with any obstacles regardless whether they are static or dynamic ones. Collision avoidance requires another world model which at any time truly represents the distances to all real obstacles in the robot's vicinity in order to achieve reflexive behaviour. This world model should be as robust as possible against disturbing effects which may cause obstacles which do not really exist to appear in the world model. To achieve a reliable and complete world model necessitates the environment to be perceived without any gap while the demanded robustness requires some kind of filter mechanism. If the AMR is to be used in a dynamic environment this furthermore requires a realtime perception process.

The major difficulty in robot perception is to find out what the sensor signals tell about the real world - in other words: the problem is to interpret the sensor signals in such a way that each world model represents an optimal estimate of the real world's state with respect to the model's specific purpose. We feel that the design of a system for sensor signal interpretation basically is an optimization process, since on the one hand the requirements of the problem to be solved have to be considered and on the other the information which can be provided by the individual sensors. A crucial problem in sensing is that of coping with uncertainty. The synergistic use of multiple physically different sensing devices, i.e. the use of heterogenous sensors may help to overcome the shortcomings of each individual sensing device since information can be provided concerning environmental features that cannot be perceived using individual sensing devices stand alone (complementarity). Besides the fusion of redundant information from multiple sensors can reduce uncertainty especially if an individual sensor fails. To sum up, it may be said that synergism in sensing is the key to obtain more accurate world models and consequently more reliable autonomous mobile robots.

To begin with, the paper's next section presents a survey of the MOBOT-IV system architecture. Thereafter, section three gives a brief survey of related work in sonar world modeling while section four describes our approach, *Pilot Specific World Modeling* (*ps*-WM) in detail. Section five introduces our motion control method. Finally, section six presents some concluding remarks.

2. MOBOT's System Architecture

MOBOT's system architecture comprises five levels of control (Fig. 2.1). Among these, only the three upper levels require sensory input about the robot's world. *Planner* and *Navigator* perform goal-oriented tasks and consequently need either topological (Planner) or geometrical (Navigator) information about the environment. The Pilot's task is to perform collision avoidance, i.e. the *Pilot* ensures reflexive behaviour. Concerning the realtime conditions an autonomous mobile robot has to cope with, it may be stated that the lower a component is located in the

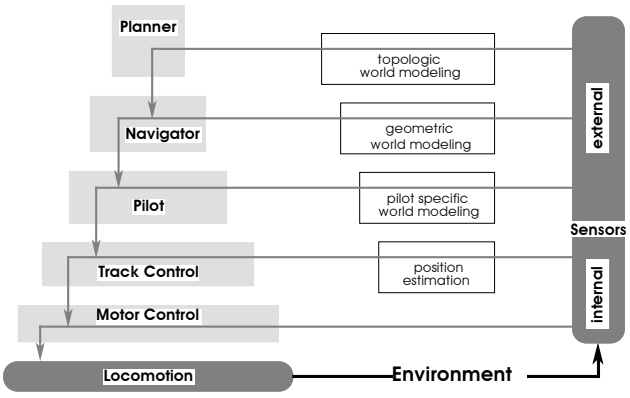

Fig. 2.1: MOBOT's fundamental system architecture.

hierarchical control structure the more it has to take into account realtime aspects. This means that the associated information-producing components have to perform their computations in realtime, too.

Actually, the sensor signal interpretation is based on two sensory sources: a laserradar system and a sonar sensor system which consists of 24 Polaroid ultrasound-transducers. The laserradar system is mounted on top of the vehicle and consists of four triangulation-based laser rangefinders. When operating, the whole unit is revolving once per second and delivers four panoramic scans per revolution with 720 range readings per scan. These are used to perform two different tasks: statistical interpretation and geometrical interpretation (Primary Feature Extraction, PFE). The statistical interpretation provides information about the orientation and position of the vehicle in its indoor environment. The geometrical interpretation uses the range readings to compute a line description of the AMR's actual environment which is called Radar Map (RM) [6].

3. Related Work in Sonar World Modeling

Beckerman & Oblow have developed a rule-based approach which deals with the treatment of systematic errors [1]. Each cell of their grid is either labeled as *empty*, *occupied*, *unknown*, or *conflict*. Conflicts may occur whenever an object is being observed at different times from different locations. The approach has disadvantages concerning dynamic obstacles, and moreover is time-consuming since they use a two-dimensional sensor model. The probabilistic approach of Elfes & Moravec generates an occupancy grid which explicitly represents free and occupied regions in space [3]. Their sensor model is two-dimensional, as well. Since they take into account the ultrasound beam as a whole a large number of grid cells corresponding to the projection of the beam has to be updated. This approach requires quite a lot of computing time, as well. Borenstein & Koren call their method for grid-based mapping the vector field histogram [2]. They use a one-dimensional sensor model which reduces the real sound wave to the beam's acoustic axis. This trivial sensor model causes a drastic transfer of the sensing-computation ratio to the credit of the sensors and leads to surprisingly good results.

4. *ps*-WM: Pilot Specific World Modeling

The above-mentioned approaches show that sonar sensing basically suffers from three problems: specularity, beam width, and frequent misreadings due to either external ultrasound sources or crosstalks. We feel that it is just the beam width which makes sonars suitable for collision avoidance world modeling provided that specularities can efficiently be eliminated and frequent misreading can be suppressed. In the latter case each valid sonar range reading tells two things. First, there is an obstacle somewhere in the measured distance and second there is no obstacle in between since no real sound wave can go through a real object. Concerning the reflected beam it is essential that its width strongly depends on the texture of the target. The more a target absorbs the transmitted sound wave's energy the smaller is the beam width (at a given frequency). Experiments demonstrated that the maximum beam width of the Polaroid transducers is approximately 36°.

Under realtime aspects it would be desirable to operate all available sonars in parallel. Unfortunately, a crucial problem in sonar sensing with multiple sensors is that of mutual influence: a transducer's sound wave must not influence other sensors (crosstalks). Thus the individual sensors have to operate either spatially or temporally decoupled. Two or more sensors are said to be spatially decoupled if they operate in parallel without influencing each other. If two sensors operate sequentially so that the sound wave of the first sensor's transducer has lost so much energy that it cannot affect the second sensor they are said to be temporally decoupled. Thus, sonar sensing for collision avoidance requires a compromise between the physical constraints of sound on one hand and the realtime requirements on the other. We feel that it is a good compromise to operate groups of sonars sequentially, i.e. the groups are temporally decoupled while the individual sensors of each group are spatially decoupled. The maximum number of sensors of a group depends on their individual distances and the angles they have with respect to each other and thus on the physical dimensions of the AMR. In order to obtain a panoramic scan of the environment two arbitrary but adjacent transducers are rotated 15° with respect to each other. Another parameter which affects the maximum number of sensors of a group is the desired maximum range. Depending on this parameter we operate the sonars either in 4 groups of 6 sensors each or in 6 groups with each group consisting of 4 sensors.

Our world model for collision avoidance is a polar coordinate system called Sector Map (SCM) which is defined locally to the AMR. Since the sector map consists of 48 sectors each individual sector covers an angle of 7.5° and represents the distance between the AMR's boundary and an obstacle. As a consequence of misreadings caused by

specular reflections the direct use of the sonar range readings to update the SCM is doomed to failure. Thus, the (grid-based) accumulation of partially redundant sonar range readings over time is a necessary precondition in order to achieve a reliable sector map. The SCM serves as the input of our pilot component which performs collision avoidance. Consequently, we call our approach Pilot Specific World Modeling (*ps*-WM). *ps*-WM is based on the method of Borenstein & Koren. However, we performed three modifications:

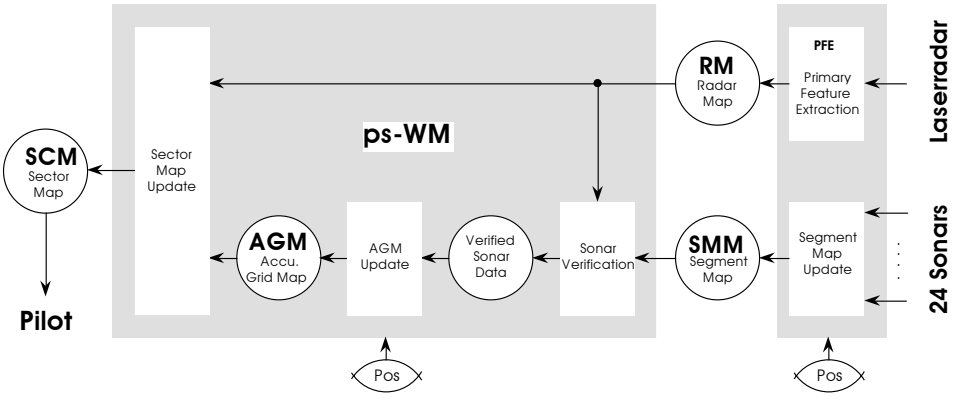

Fig. 4.1: The basic structure of Pilot Specific World Modeling.

- We verify the actual sonar range readings with data obtained from our laserradar.
- The verified sonar range readings are accumulated over a fixed distance based on a trivial, zero-dimensional sensor model.
- To update the SCM the sonar data are interpreted using a simple stochastic model.

Thus, we use sensor data of both, the laserradar and the sonar sensor system in order to update the SCM. Since all individual sonar range readings correspond to the vehicle's internal coordinate system they are transformed into a global cartesian coordinate system called Segment Map (SMM) in order to become independent from the AMR's motion. Then, the SCM-update is performed in three computational steps (Fig. 4.1).

1. step. The sonar range readings are being verified using laserradar data (RM). Three basic situations are considered:

1. The sonar information is far beyond the information obtained by the laserradar. In this case the sonar information is being ignored, resulting in a very efficient elimination of specular reflections.
2. Whenever sonar range information is in the close vicinity of laser-based range information (with respect to a certain tolerance) both sensors obviously have detected the same obstacle. As a consequence of the laserradar's high angular resolution the sonar information is being ignored, too.
3. Laser range information is far beyond sonar range information. In this case an obstacle has been detected by the sonars but not by the laserradar. Consequently the laser range information is being ignored.

A positive result of this verification is that e.g. an open door is detected much earlier than if sonar data alone were used.

2. step. During the second step the verified sonar data are incrementally integrated into a grid-based intermediate representation, called Accumulative Grid Map (AGM) which has a size of 100 * 100 quadratic cells. If the cell size (which is a parameter) is set to 10 cm this results in an AGM size of 10 * 10 m. Note that the vehicle is always located in the center of the AGM. Its purpose is to locally and spatially stabilize the sonar range readings as well as to suppress disturbing effects and thus reduce uncertainty. Although the AGM is defined relatively to the AMR it represents the distance information in global coordinates. Thus, the AGM does not follow the vehicle's rotational movements: it keeps its orientation with respect to the environment.

The AGM is periodically updated using a zero-dimensional sensor model. Differently from Borenstein, we do not use the information about free space associated with each sonar range reading to revise already occupied grid cells. If we did that, specular reflections could cause correctly occupied grid cells to be attributed as being empty. Instead, while the AMR is moving all valid sonar range readings produced over a fixed distance are integrated in the AGM. Experiments have proved a distance of 1m to be appropriate. Consequently, all stored range readings which are older than '1 m' are being removed from the AGM. Moving obstacles can leave traces even while the vehicle is not moving. Therefore, all stored sonar range readings which are older than a certain period of time are also being removed. This provision prevents the vehicle from being blocked.

In order to suppress random disturbances caused by crosstalks or other ultrasound sources we use an evaluation method which takes into account how frequent a quadratic cluster of grid cells is affected by the sonar range readings during an accumulation period. Every cell of the grid has an associated weight and is taken into account for updating the SCM only if the weight exceeds a certain threshold.

3. step. The AGM and the RM are complementary to each other since the AGM represents information which is not represented by the RM, and vice versa. Thus, both maps reflect the complementary physical properties of the laserradar and the sonar sensor system, respectively. The third step uses the actual AGM and the actual RM in order to

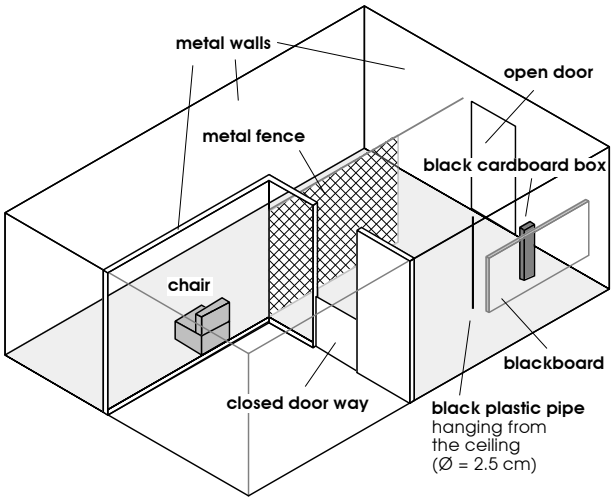

Fig. 4.2: Our test lab.

update the sector map. Each sector's corresponding laser based range value is being compared with the associated sonar based range value and the smaller value is used to update the sector. Sectors representing a range value are called occupied sectors.

Since we use a trivial sonar sensor model which does not take into account a sonar's beam width, the maximum position error of an obstacle is half the beam width. For the sake of security we copy the sonar range value of an occupied sector into adjacent sectors. An adjacent sector which is already occupied by sonar range data remains unchanged because of the very high probability that the range data of this sector are correct.

If we fire the 24 sonars in four groups of six sensors each it needs 200 ms to achieve a panoramic scan. To update the SCM we need 150 ms.

All experiments were performed using the real robot. Our test lab has metal walls and was additionally equipped with a metal fence, a thin plastic pipe and a black cardboard box as shown in Fig. 4.2. Fig. 4.3 shows screen dumps of a test run. MOBOT-IV was moving along the dotted track. Fig. 4.3a shows the Radar Map corresponding to the vehicle's final position. Neither the metal fence nor the pipe nor the box were detected by the laserradar. Fig. 4.3b shows the Accumulative Grid Map which has been built up during the last meter of travel. Without the sonar information, the vehicle would have been unable to detect the obstacles mentioned previously. Fig. 4.3c shows the resulting Sector Map. Please note that the door in front of the robot is perceived to be open from a distance of more than 2 meters.

5. Motion Control

As already mentioned, the sector map serves as input of a pilot component. The Pilot transforms the SCM's distance values into obstacle forces and fuses these to a virtual position deviation Δvp. The basic idea of the algorithm is to perform track control using Δvp instead of the real po-

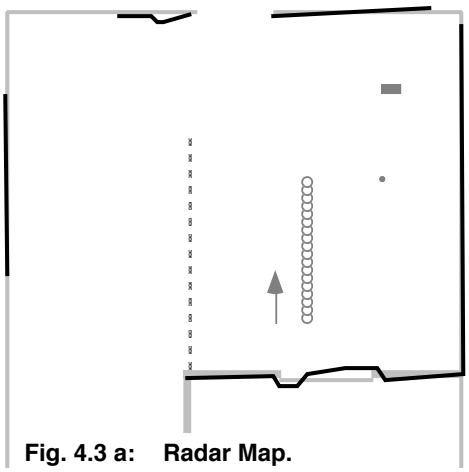

Fig. 4.3 a: Radar Map.

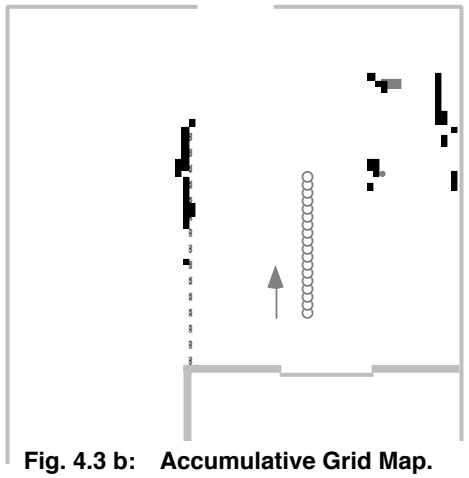

Fig. 4.3 b: Accumulative Grid Map.

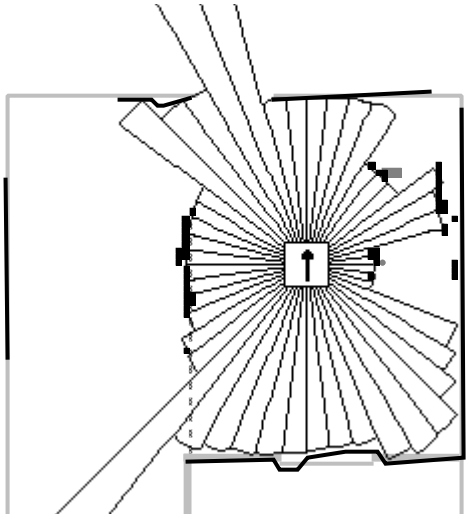

Fig. 4.3 c: Sector Map.

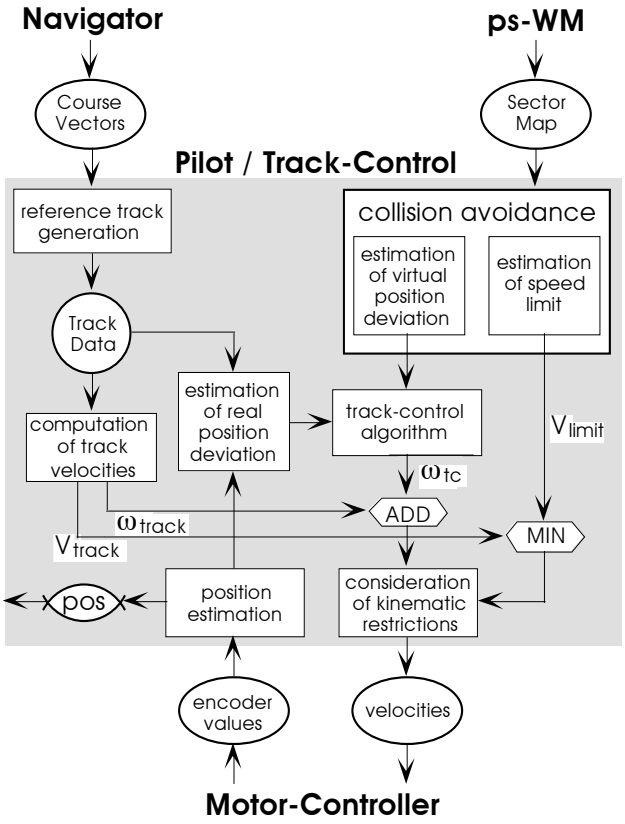

Fig. 5.1: The basic structure of Pilot/Track-Control.

sition deviation to maneuver around obstacles. Thus, both control loops shown in Fig. 2.1 are performed by one component (Fig. 5.1).

The task of the Pilot, as part of our cascaded motion control, is to avoid collisions with obstacles which were unknown while the Navigator was planning a collision free course [5]. The Pilot responds to obstacles in the close vicinity of the vehicle only if they are either unknown, mobile, or have a different position. However, the Pilot's basic goal is to follow the Navigator's course. Since we believe that it is difficult to perform our motion control-philosophy by using a known collision avoidance approach like the 'Generalized Potential Field Method' [7] or 'Dynamic Path Planning' [8] we have developed a new concept.

The orientation of the sector map's polar axis is identical to the orientation of the global cartesian coordinate system's x-axis. Thus, the SCM has to be rotated such that the orientation of its polar axis becomes identical to the orientation of the AMR's longitudinal axis. Please note that the origin of the SCM is located at the center of the AMR, while the center of its rear axis is the origin of the robot's local coordinate system. Moreover, it is optionally possible to take into consideration the AMR's motion in the immediate future by additionally rotating the sector map's polar axis in dependence on the AMR's actual angular velocity.

Naturally, the potential danger to collide with an obstacle

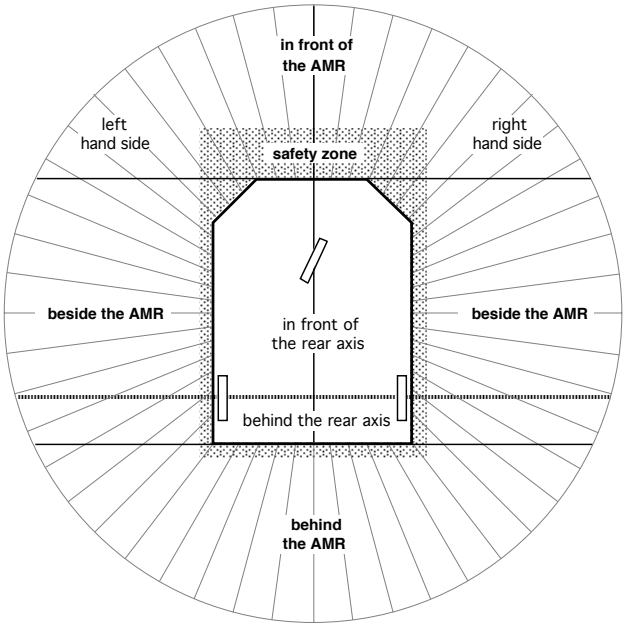

Fig. 5.2: Zones and corresponding sectors.

increases with decreasing obstacle distances. In case of a static obstacle, the relative velocity of this obstacle solely depends on the motion of the robot. Assume that the vehicle is moving forward. Then the danger of a collision is much greater if an obstacle exists in front of the vehicle than if an obstacle is beside the AMR. By contrast, there is no danger, whenever an obstacle is behind the robot. While the robot is moving, changes of distances to obstacles on the vehicle's left-hand side are complementary to changes of distances to obstacles on the right-hand side of the vehicle. Changing obstacle distances in front and behind the vehicle's rear axis are complementary, too. To handle these different cases, the sector map is partitioned into several groups (Fig. 5.2).

The groups 'left' and 'right' are given by the orientations of the individual sectors. Each individual sector's distance value is used to compute the x-distance and the y-distance with respect to the robot's safety zone and local coordinate system (Fig. 5.3). Fig. 5.2 shows how the x-distance is used to separate the sectors into four groups:

- $\mathcal{F}$: in front of the vehicle
- $\mathcal{bf}$: beside the vehicle, in front of the rear axis
- $\mathcal{bb}$: beside the vehicle, behind the rear axis
- $\mathcal{B}$: behind the vehicle.

Additionally, if a distance value indicates that an obstacle has entered the vehicle's safety zone the AMR has to immediately perform an emergency stop. In this case the Navigator has to compute a new course.

In order to compute the obstacle forces each group of sectors uses an individual strategy to weight the distance values (Fig. 5.3):

- Obstacles behind the vehicle are being ignored.
- For each sector in all the other groups, an obstacle

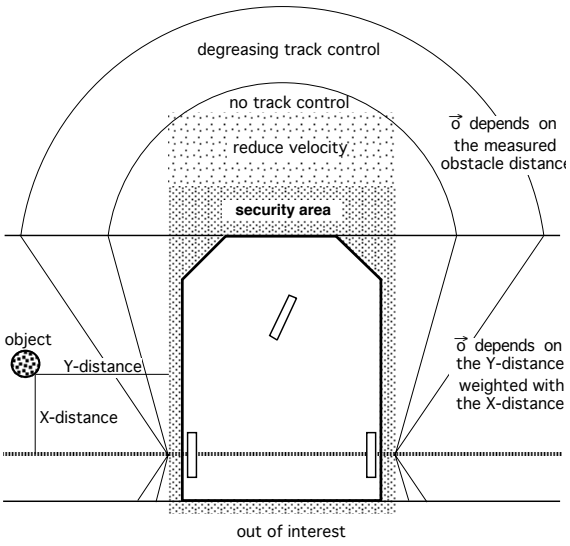

Fig. 5.3: Weighting scheme for distance values.

force $\vec{o}(d)$) is computed, with $\vec{o}(d)$ being a hyperbolic function. $\vec{o}(d)$ is positive for each sector of group 'left' and negative otherwise.

- Concerning the sectors of group $\mathcal{F}$, the argument d of $\vec{o}(d)$ becomes the measured distance value. Additionally, the x-distance is used to compute a speed limit depending on the minimum stopping distance.
- Concerning the sectors of the groups $\mathcal{bf}$ and $\mathcal{bb}$, $\vec{o}(d)$ depends on the y-distance weighted with the associated x-distance such that $|\vec{o}|(d)$ becomes proportional to the x-distance. Especially, the obstacle forces of obstacles located somewhere along the virtual elongation of the robot's rear axis (x-distance = 0) become equal to zero. Since the x-distances of all obstacles behind the elongation of the robot's rear axis are negative, the obstacle forces of group *bb's* sectors are complementary to the obstacle forces of group *bf's* sectors. The reason for this provision is to take the relative velocity of obstacles into account whenever an obstacle is located at the vehicle's side and the robot turns towards the obstacle's direction.

The next step uses the obstacle forces to compute the virtual position deviation. Actually, we have implemented three alternative fusion methods:

Method 1 adds the maximum obstacle force of group 'left' and the maximum obstacle force of group 'right' to guarantee that both, small and big obstacles have the same weight. In this case the vehicle will always pass a notch in its middle. However in practice, it is often better to pass a corridor nearby a small obstacle, since e.g. a long wall limits the freedom of movement much more than a flower-pot. Using this method it is furthermore impossible to respond to an obstacle in front of the AMR in time, if there is a small obstacle quite near beside the robot. The major disadvantage of this method however is the discontinuous change of the virtual position deviation which results in a permanent oscillating motion.

Method 2 sums up all obstacle forces resulting in a more continuous change of the virtual position deviation Δvp. Using this method the vehicle will respond to big obstacles in front of the vehicle in time. However, if many sectors represent very short obstacle distances, the robot responds too late to small obstacles which are represented in one or two sectors, only. Thus, the AMR may pass a single, small obstacle located on one side of a corridor at too close a distance, if the corridor's other border (e.g. a wall, a long cupboard etc.) is very long.

Method 3 uses the arithmetic mean of both methods described above. This combines the advantages of both methods. In comparison to method 2, small obstacles in the close vicinity of the AMR are considered more strongly. The weight of a large obstacle on one side of a corridor is higher in comparison to the first method. Furthermore the smoothness of Δvp can be increased. However, as any compromise, this method cannot be optimal in all cases.

If Δvp exceeds a certain threshold, it serves as an additional input of our Track-Control algorithm. If there are no unknown obstacles in the AMR's vicinity, this algorithm tries to follow a precomputed track which consists of mathematical descriptions of lines, arcs and steering-phases. The latter ones are described as curvature continuous curves and depend on the kinematic constraints of the vehicle (e.g.: clothoids for an AMR with two independently driven rear wheels). Let V be the linear velocity and ω the angular velocity of the vehicle. Theoretically it is possible to follow this precalculated track without any deviation. This could be achieved by executing a (V,ω) command by the motor-controller, with (V,ω) depending on the actual curvature of the track and the actual speed. In practice, there are certain deviations $(\Delta pos,\Delta\varphi)$ resulting from a slippery or unsmooth floor, the duration of a program cycle, uncertainties in measuring the motion parameters, etc. To control these deviations is task of the Track-Control component. This is done by adding an offset $\omega_{tc}(\Delta pos,\Delta\varphi)$ to ω_{track}.

Note that the Navigator, according to its geometric world model, guarantees that the course vectors define a collision free course. Thus, the precomputed track is indeed collision free as long as the robot's environment remains unchanged. Collision avoidance is required only if unknown or moving obstacles appear. Our approach uses the Track-Control algorithm to maneuver around such obstacles by treating the virtual position deviation as if it were a real position deviation. Let's assume that an obstacle is e.g. located on the vehicle's left hand side. This leads to a positive value of Δvp, which simulates a position deviation to the left. Consequently, the Track-Control algorithm computes a negative offset $\omega_{tc}(\Delta vp,0)$ and causes the robot to follow a path, which is a balance between both, real and virtual position deviation. Consequently, the robot leaves the precomputed track to its right-hand side until the obstacle has been passed. Then, Δvp becomes equal to zero again which causes Track-Control to

maneuver the AMR back to the precomputed track again.

The major problem of this collision avoidance method is to achieve a stable balance between both ω-offsets ($\omega_{tc}(\Delta pos,\Delta\varphi)$ and $\omega_{tc}(\Delta vp,0)$) such that the distance between the robot and an obstacle is the same whenever the AMR passes an obstacle. If $\omega_{tc}(\Delta pos,\Delta\varphi)$ increases, the vehicle tries to maneuver back to the precomputed track until $\omega_{tc}(\Delta vp,0)$ has a complementary value. As a result, the robot passes an obstacle more closely if the track deviation increases. To eliminate this effect, $\omega_{tc}(\Delta pos,\Delta\varphi)$ is continuously decreased whenever Δvp exceeds a certain threshold (Fig. 5.3). This provision is allowed, since collision avoidance has the highest priority if an obstacle is in the robot's close vicinity. In a dangerous situation like this it is absolutely unimportant to care about the precomputed track. Now you can think of special environments, where it is impossible for the robot to reach the destination of the track. Thus, if Δpos exceeds a given threshold, the Pilot assumes that it is impossible to follow the precomputed track. In this case the Pilot stops the vehicle and asks the Navigator to compute new course vectors. They describe a new collision free course, which is transformed into an equivalent track description.

ω_{track} and V_{track} depend on both, the actual curvature of the track and the robot's actual speed. The computed ω is the sum of ω_{track}, $\omega_{tc}(\Delta pos,\Delta\varphi)$, $\omega_{tc}(\Delta vp,0)$. Due to the vehicle's kinematic constraints it may be necessary to reduce V in order to achieve a certain value of ω. Furthermore, if V has already been limited by the collision avoidance algorithm described above, it may be necessary to decrease ω_{track} in order to be able to follow the track's curvature. Thus, the dependencies of V and ω are considered in the algorithm's last step.

The collision avoidance algorithm has been intensively tested using our simulation environment. The simulated AMR had to perform several tasks, e.g. wallfollowing, driving along a corridor, driving into a dead end, and passing an open door. Fig. 5.4 shows a simple scene edited in our simulation environment. It is assumed that the Navigator has computed a collision free course along the corridor, entering a new room and passing a table and two chairs. The associated track is indicated by the black line. Unfortunately, somebody has moved the table and the chairs and additionally put two boxes into the corridor. The dotted line indicates the real track, which has been computed by the Pilot using method 2 and a simulated SCM. This results in a short distance between the track and the left side of the door frame.

Using the real machine's computer system (i.e. in the absence of simulation and screen display routines) the algorithm's real time constraints of 16 computation cycles per second are fulfilled.

6. Conclusions

A 3-step approach for collision avoidance world mode ling is presented first. The approach is based on sensor data obtained by a revolving laser range finder and a sonar sensor system. Experiments have tested the method under realtime conditions and shown that it is a considerable improvement of sonar sensing for collision avoidance since it effectively suppresses specular reflections. The motion control approach presented uses a virtual position deviation as input for a track control algorithm to aviod collisions. A method for transforming the pilot specific world model into obstacle forces and their fusion into a virtual position deviation is described. Currently, we are integrating and testing *ps*-WM and the Pilot/Track-Control component on our mobile test-platform MOBOT-IV.

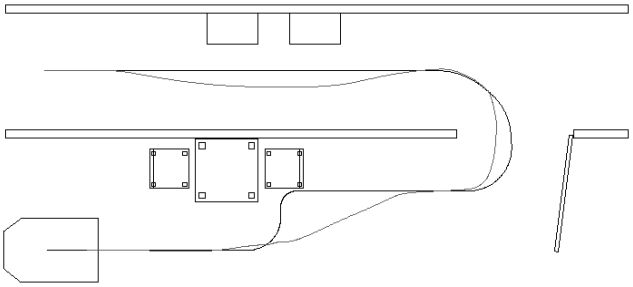

Fig. 5.4: Example track.

References

[1] Beckerman, M., Oblow, E.M.: *Treatment of Systematic Errors in the Processing of Wide-Angle Sonar Sensor Data for Robotic Navigation*, IEEE Transactions on Robotics and Automation, Vol. 6, NO. 2, April 1990

[2] Borenstein, J., Koren, Y.: *Real-time map-building for fast mobile robot obstacle avoidance*, SPIE Vol. 1388 Mobile Robots V (1990)

[3] Elfes, A.: *Dynamic Control of Robot Perception Using Stochastic Spatial Models*, Int. Workshop on Information Processing in Autonomous Mobile Robots, Munich, Germany, 1991

[4] Flynn, A.M.: *Combining Sonar and Infrared Sensors for Mobile Robot Navigation*, The International Journal of Robotics Research,Vol. 7, No. 6, 1988

[5] Hoppen, P., Knieriemen, T., von Puttkamer, E.: *Laser-Radar based Mapping and Navigation for an Autonomous Mobile Robot*, Proc. of the IEEE Int. Conf. on Robotics and Automation, Cincinnati, Ohio, 1990

[6] Knieriemen, T., von Puttkamer, E., Roth, J.: *Extracting Lines, Circular Segments and Clusters from Radar Pictures in Real Time for an Autonomous Mobile Robot*, Euromicro '91, Workshop on Real-Time Systems, Paris-Orsay, 1991

[7] Krogh, B.H.: *A Generalized Potential Field Approach of Obstacle Avoidance Control,* Robotics Research: The Next Five Years and Beyond, Bethlehem, Pennsylvania, August 14-16, 1984

[8] Lumelsky, V., Stepanov, A.: *Path Planning strategies for a Point Mobile Automaton Moving Amidst Unknown Obstacles of Arbitrary Shape,* Algorithmica 1987, 2: 403-430

[9] Matthies, L., Elfes, A.: *Probabilistic Estimation Mechanisms and Tesselated Representations for Sensor Fusion*, SPIE Sensor Fusion: Spatial Reasoning and Scene Interpretation, 1988